

Teoría de Lenguajes  
Preguntas de examen final  
27 junio 2022

**Ejercicio 1.** Dar un algoritmo que decida si dos lenguajes regulares son iguales. Justificar la correctitud. Analizar la complejidad.

**Ejercicio 2.** Dada una gramática libre de contexto  $G$ , existe una constante  $c$  demostrar que para todo par de símbolos no terminales  $A, B$ , para toda cadena de terminales  $w$ , para toda cadena de terminales y no terminales  $\alpha$ ,

$$\text{si } A \xRightarrow[L]{i} wB\alpha, \text{ entonces } i \leq c^{|w|+2}.$$

**Ejercicio 3.** Dar dos algoritmos distintos para determinar si el lenguaje aceptado por un autómata finito dado es el conjunto de todas las cadenas del alfabeto. Justificar cada uno y dar su complejidad algorítmica.

**Ejercicio 4.** Dar un algoritmo que determine si un lenguaje regular dado es infinito. Justificar y dar la complejidad del algoritmo.

**Ejercicio 5.** ¿Cuántos autómatas finitos deterministas con dos estados pueden construirse sobre el alfabeto 0,1?

¿Cuántos autómatas finitos no determinísticos con dos estados pueden construirse sobre el alfabeto 0,1?

¿Cuántos autómatas de pila con dos estados pueden construirse con alfabeto de entrada  $A$ , alfabeto de pila  $Z$ , y máximo cantidad de símbolos en cada transición  $M$ ?

**Ejercicio 6.** Fijados los alfabetos  $\Delta$  y  $\Gamma$ , ¿Cuántos autómatas de pila distintos  $(Q, \Sigma, \Delta, \Gamma, q_0, F)$  determinísticos hay, Si  $Q$  tiene 5 estados y en cada transición se escriben en la pila 0,1 o 2 símbolos? ¿Y cuántos no determinísticos?

**Ejercicio 7.** Definir cuando una gramática libre de contexto es recursiva a derecha. Dar el algoritmo de eliminación de recursión a derecha (inmediata y no inmediata), su justificación de correctitud, y su complejidad computacional. Dar un ejemplo.

**Ejercicio 8.** Dar el algoritmo de pasar a forma normal 3-chomsky. Justificar correctitud y dar la complejidad computacional.

$$A \rightarrow a$$

$$A \rightarrow BC$$

$$A \rightarrow BCD$$

No se permiten producciones  $A \rightarrow B$ .

donde  $A, B, C, D$  son no terminales y  $a$  es terminal.

Entonces son 3-Chomsky

$$S \rightarrow ABC$$

$$A \rightarrow BDE$$

$$A \rightarrow a$$

$$A \rightarrow BC$$

No son 3-Chomsky

$$A \rightarrow B$$

$$A \rightarrow ABCDE \text{ tampoco es 3-Chomsky}$$

$$A \rightarrow abcdef \text{ tampoco es 3-Chomsky}$$

**Ejercicio 9.** Consideremos el transductor finito dado por una máquina de Mealy  $(S, S_0, \Sigma, \Gamma, T, G)$

,

$S$  es un conjunto finito de estados

$S_0$  es un estado inicial

$\Sigma$  es el alfabeto de entrada

$\Gamma$  es el alfabeto de salida

$\delta : S \times \Sigma \rightarrow S$  es la función de transición

$\gamma : S \times \Sigma \rightarrow \Gamma$  mapea un estado y un símbolo de entrada a un símbolo de salida

Adaptar el algoritmo de minimización de autómatas finitos a una minimización de máquina Mealy.

Ayuda: Definir la relación de equivalencia considerando la función  $\delta$  extendida y la función gamma extendida.

**Ejercicio 10.** Demostrar que dada una gramática regular a derecha se puede obtener una gramática regular a izquierda equivalente. Tener en cuenta que disponemos del algoritmo para ir de gramática regular a derecha a autómata finito y que también disponemos del algoritmo para ir de autómata finito a gramática regular a derecha.

Ayuda: hallar la gramática del reverso de un lenguaje y el autómata finito del reverso de un lenguaje, o sea, dada  $G$  tal que  $L = L(G)$  hallar  $GR$  tal que  $LR = L(GR)$  y dado  $M$  tal que  $L = L(M)$  hallar  $MR$  tal que  $LR = L(MR)$ , donde  $LR$  es el reverso del lenguaje  $L$ .

Ayuda adicional: Hacer un ejemplo de gramática con 2 no terminales y 2 terminales y que genere una sola cadena.

Posible ayuda adicional: invertir producciones y transiciones.

**Ejercicio 11.** Dado un autómata finito no determinístico  $A$  pero sin transiciones lambda dar un algoritmo que construye el autómata finito no determinístico que acepta el lenguaje  $L(A)L(A)^R$ . Demostrar por inducción en el largo de las cadenas de que el algoritmo es correcto. Determinar la complejidad computacional del algoritmo.

**Ejercicio 12.** Definimos un autómata de pila de doble entrada como una tupla  $P = (Q, \Sigma, \tau, \Gamma, \delta, q_0, Z_0, F)$  donde

$$\delta : Q \times \Sigma \cup \{\lambda\} \times \tau \cup \lambda \times \Gamma \rightarrow \mathcal{P}(Q \times \Gamma^*).$$

La función  $\delta$  es tal que las transiciones lambda ocurren en ambas cintas a la vez. Es decir, no hay transiciones que lean de una cinta y no de la otra.

Demostrar que para todo autómata de pila de doble entrada que acepta por estado final siempre se puede encontrar otro equivalente que acepta por pila vacía.

**Ejercicio 13.** Dado un autómata de pila  $P$  determinístico dar el autómata de pila no determinístico que acepta  $L' = \{wv^R : w, v \in L(P)\}$  donde  $w^R$  es la reversa de  $w$ , es decir si  $w = abcw^R = cba$ .

**Ejercicio 14.** Mostrar que si  $L$  es un lenguaje aceptado por un autómata de pila determinístico por pila vacía entonces ninguna palabra de  $L$  es prefijo propio de otra palabra de  $L$ . ¿Vale lo mismo en caso de que el autómata de pila sea no-determinístico?

**Ejercicio 15.** Dado  $R$  un lenguaje regular, y dado  $L$  un lenguaje libre de contexto determinístico, ¿Es decidible si  $L=R$ ? Es decir, ¿hay un algoritmo capaz de decidir la igualdad? En caso de que sí, dar tal algoritmo y justificar. En caso de que no, dar la demostración de indecidibilidad.

Respuesta: Teorema 10.6 Hopcroft 1976

**Ejercicio 16.** Dado  $R$  un lenguaje Regular, y dado  $L$  un lenguaje libre de contexto determinístico, ¿Es decidible si  $R$  está incluido en  $L$ ?

Respuesta: Teorema 10.6 Hopcroft 1976

**Ejercicio 17.** Dadas dos expresiones regulares  $\alpha$  y  $\beta$ , dar un método para comprobar si son o no equivalentes. Justificar la correctitud. Dar la complejidad computacional

**Ejercicio 18.** Dar un algoritmo que transforma una gramática  $LL(k)$  en otra  $LL(k)$  de la forma

$$A \rightarrow a\alpha \text{ y } A \rightarrow \lambda.$$

**Ejercicio 19.** Considerar la gramática  $S \rightarrow SS|a$

a) Contar la cantidad de árboles de derivación más a la izquierda de  $a^n$

b) Contar la cantidad de árboles de derivación más a la derecha de  $a^n$

AYUDA: Ensayar con  $n = 1, n = 2, n = 3, n = 4, n = 5$  hasta conseguir la forma general.

**Ejercicio 20.** Dar un algoritmo que transforme cada gramática libre de contexto  $G$  sin producciones  $A \rightarrow \lambda$  en otra  $G'$  que reconoce el mismo lenguaje pero tal que cada producción es de la forma  $A \rightarrow a\alpha$ , con  $a$  un símbolo terminal y  $\alpha$  una cadena de no-terminales.

Ayuda: Definir un orden parcial  $<$  entre los no terminales de  $G$  donde si  $A \rightarrow B\beta$  entonces  $A < B$ . Iterar cambiando la gramática donde cada producción con cabeza  $A$  tenga un cuerpo que sea un terminal o con un no-terminal  $B$  donde  $A < B$ .

Justificar la correctitud y dar la complejidad del algoritmo.

**Ejercicio 21.** Dar un algoritmo que transforme cada gramática libre de contexto  $G$  sin producciones  $A \rightarrow \lambda$  en otra  $G'$  que reconoce el mismo lenguaje pero es tal que ninguna producción tiene un lado derecho con dos no-terminales seguidos.

Ayuda: Pasar primero a forma normal de Chomsky Justificar la correctitud y dar la complejidad del algoritmo.

**Ejercicio 22.** Dar un algoritmo que transforme cada gramática libre de contexto  $G$  en otra  $G'$  que reconoce el mismo lenguaje pero es tal que si  $X_1 \dots X_k$  es el lado derecho de una producción entonces todos los símbolos  $X_1, \dots, X_k$  son distintos. Justificar la correctitud y dar la complejidad del algoritmo